```
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSSSSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSSSSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSSSSS
RRR         RRR  MMMMMM   MMMMMM  SSS
RRR         RRR  MMMMMM   MMMMMM  SSS
RRR         RRR  MMMMMM   MMMMMM  SSS
RRR         RRR  MMM   MMM   MMM  SSS
RRR         RRR  MMM   MMM   MMM  SSS
RRR         RRR  MMM   MMM   MMM  SSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSS
RRR   RRR        MMM         MMM              SSS
RRR   RRR        MMM         MMM              SSS
RRR   RRR        MMM         MMM              SSS
RRR      RRR     MMM         MMM              SSS
RRR      RRR     MMM         MMM              SSS
RRR      RRR     MMM         MMM              SSS
RRR         RRR  MMM         MMM  SSSSSSSSSSSS
RRR         RRR  MMM         MMM  SSSSSSSSSSSS
RRR         RRR  MMM         MMM  SSSSSSSSSSSS
```

```
**FILE**ID**RMSOENTER                                   C 12




RRRRRRRR    MM      MM    SSSSSSSS    000000    EEEEEEEEEE  NN      NN  TTTTTTTTTT  EEEEEEEEEE  RRRRRRRR
RRRRRRRR    MM      MM    SSSSSSSS    000000    EEEEEEEEEE  NN      NN  TTTTTTTTTT  EEEEEEEEEE  RRRRRRRR
RR      RR  MMMM  MMMM  SS          00      00  EE          NN      NN      TT      EE          RR      RR
RR      RR  MMMM  MMMM  SS          00      00  EE          NN      NN      TT      EE          RR      RR
RR      RR  MM  MM  MM  SS          00    0000  EE          NNNN    NN      TT      EE          RR      RR
RR      RR  MM  MM  MM  SS          00    0000  EE          NNNN    NN      TT      EE          RR      RR
RRRRRRRR    MM      MM    SSSSSS    00  00  00  EEEEEEEE    NN  NN  NN      TT      EEEEEEEE    RRRRRRRR
RRRRRRRR    MM      MM    SSSSSS    00  00  00  EEEEEEEE    NN  NN  NN      TT      EEEEEEEE    RRRRRRRR
RR  RR      MM      MM          SS  0000    00  EE          NN    NNNN      TT      EE          RR  RR
RR  RR      MM      MM          SS  0000    00  EE          NN    NNNN      TT      EE          RR  RR
RR    RR    MM      MM          SS  00      00  EE          NN      NN      TT      EE          RR    RR      ....
RR    RR    MM      MM          SS  00      00  EE          NN      NN      TT      EE          RR    RR      ....
RR      RR  MM      MM  SSSSSSSS      000000    EEEEEEEEEE  NN      NN      TT      EEEEEEEEEE  RR      RR    ....
RR      RR  MM      MM  SSSSSSSS      000000    EEEEEEEEEE  NN      NN      TT      EEEEEEEEEE  RR      RR    ....


LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
   (3)      85       DEFINITIONS
   (4)     111       RMS$ENTER, Enter File in Directory
   (5)     206       RM$RECOVER_FWA, Recover FWA from Expanded String
```

```
0000            $BEGIN  RMS0ENTER,000,RMS$RMS,<ENTER FILE IN DIRECTORY>
0000
0000   ;
0000   ;*****************************************************************************
0000   ;*                                                                           *
0000   ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000   ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000   ;*  ALL RIGHTS RESERVED.                                                     *
0000   ;*                                                                           *
0000   ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0000   ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
0000   ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
0000   ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0000   ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
0000   ;*  TRANSFERRED.                                                             *
0000   ;*                                                                           *
0000   ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
0000   ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
0000   ;*  CORPORATION.                                                             *
0000   ;*                                                                           *
0000   ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
0000   ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000   ;*                                                                           *
0000   ;*                                                                           *
0000   ;*****************************************************************************
0000   ;
```

```
0000    ;++
0000    ; FACILITY: RMS32
0000    ;
0000    ; ABSTRACT:
0000    ;       This is the highest level routine to perform the $ENTER function
0000    ;
0000    ; ENVIRONMENT:
0000    ;       VAX/VMS
0000    ;
0000    ; AUTHOR:
0000    ;       Tim Halvorsen   Aug-1979
0000    ;
0000    ; MODIFIED BY:
0000    ;
0000    ;       V03-011 DGB0058         Donald G. Blair         25-Jun-1984
0000    ;               Fix RM$RECOVER_FWA so that rather than include the
0000    ;               default directory spec for files on tape, it gives such
0000    ;               files a "[]" directory spec.
0000    ;
0000    ;       V03-010 RAS0295         Ron Schaefer            17-Apr-1984
0000    ;               Make RM$RECOVER_FWA setup the device as if it were
0000    ;               concealed using the NAM$T_DVI device id.
0000    ;               This makes sure that no translation of the name is needed.
0000    ;
0000    ;       V03-009 JEJ0026         J E Johnson             11-Apr-1984
0000    ;               Tie off invalid network operations.
0000    ;
0000    ;       V03-008 RAS0288         Ron Schaefer             5-Apr-1984
0000    ;               Fix RM$RECOVER_FWA for explicit rooted directories.
0000    ;               Note that the filespec pattern passed to the ACP
0000    ;               is in the caller's address space.
0000    ;
0000    ;       V03-007 DGB0030         Donald G. Blair         19-Mar-1984
0000    ;               Implement the XAB$V_PROPAGATE bit.
0000    ;
0000    ;       V03-006 DGB0021         Donald G. Blair         06-Mar-1984
0000    ;               Use full-length fib in order to support access mode
0000    ;               protected files.
0000    ;
0000    ;       V03-005 RAS0262         Ron Schaefer             6-Mar-1984
0000    ;               Change magtape specs to include a null "[]" directory;
0000    ;               add symbol definitions.
0000    ;
0000    ;       V03-004 RAS0219         Ron Schaefer             8-Dec-1983
0000    ;               Fix FWA constant and probe the NAM block correctly.
0000    ;
0000    ;       V03-003 KBT0584         Keith B. Thompson       12-Aug-1983
0000    ;               Clean up fwa constants
0000    ;
0000    ;       V03-002 KBT0564         Keith B. Thompson       25-Jul-1983
0000    ;               Make rm$recover_fwa work with new root directory format
0000    ;
0000    ;       V03-001 KBT0179         Keith B. Thompson       23-Aug-1982
0000    ;               Reorganize psects and rename entry point back to single '$'
0000    ;
0000    ;--
```

```
                0000    85              .SBTTL  DEFINITIONS
                0000    86
                0000    87      ;
                0000    88      ;       SYMBOL DEFINITIONS
                0000    89      ;
                0000    90
                0000    91              $DEVDEF
                0000    92              $FABDEF
                0000    93              $FIBDEF
                0000    94              $FSCBDEF
                0000    95              $FWADEF
                0000    96              $IFBDEF
                0000    97              $IODEF
                0000    98              $NAMDEF
                0000    99              $RMSDEF
                0000   100              $XABPRODEF
                0000   101
                0000   102      ;
                0000   103      ;       OWN STORAGE
                0000   104      ;
                0000   105
                0000   106      RM$ENTXAB_ARGS::
       00'10 13 0000   107              .BYTE   XAB$C_PRO,XAB$C_PROLEN_V3,XBC$C_ENTPRO
             00 0003   108              .BYTE   0
                0004   109
```

```
                        0004         .SBTTL  RMS$ENTER, Enter File in Directory
                        0004
                        0004  ;++
                        0004  ;
                        0004  ; RMS$ENTER - Enter file name in directory
                        0004  ;
                        0004  ; INPUTS:
                        0004  ;
                        0004  ;      AP = address of user argument list
                        0004  ;      Expanded name string
                        0004  ;      DVI = DEVICE NAME
                        0004  ;      DID = DIRECTORY ID
                        0004  ;      FID = FILE ID to be entered into the directory
                        0004  ;
                        0004  ; OUTPUTS:
                        0004  ;
                        0004  ;      RESULT NAME STRING
                        0004  ;--
                        0004
                        0004         $ENTRY  RMS$ENTER
                        0004
          FFF9'   30    0004         BSBW    RM$FSETI                          ; create IFAB
                        0007                                                   ; R9 = IFAB; R8 = FAB
    57    28 A8   D0    0007         MOVL    FAB$L_NAM(R8),R7                  ; get NAM address
          FFF2'   30    000B         BSBW    RM$CHKNAM                         ; check NAM validity
          61 50   E9    000E         BLBC    R0,90$                            ; branch if error
                        0011
             11   E0    0011         BBS     #NAM$V_NODE,-                     ; If this is a network
          34 A7         0013                 NAM$L_FNB(R7),-                   ; device proceed directly
             5F         0015                 NTENTER                           ; to network specific code.
                        0016
                        0016  ;
                        0016  ;      Recover FWA context from EXPANDED NAME STRING
                        0016  ;
                        0016
             63   10    0016         BSBB    RM$RECOVER_FWA                    ; recover FWA context
          57 50   E9    0018         BLBC    R0,90$                            ; branch if error
                        001B
                        001B  ;
                        001B  ;      Call ACP to enter the file name
                        001B  ;
                        001B
    5C    E2 AF   9E    001B         MOVAB   RM$ENTXAB_ARGS,AP                 ; ap = argument for xab-scan
          FFDE'   30    001F         BSBW    RM$XAB_SCAN                       ; handle pro xab
          4D 50   E9    0022         BLBC    R0,90$
10 AA   0040 8F   3C    0025         MOVZWL  #FIB$C_LENGTH,FWA$Q_FIB(R10)      ; setup FIB descriptor
        01F4 CA   9E    002B         MOVAB   FWA$T_FIBBUF(R10),-
          14 AA         002F                 FWA$Q_FIB+4(R10)
        0220 CA   B4    0031         CLRW    FIB$W_VERLIMIT+FWA$T_FIBBUF(R10); assure version limit = 0
                  3C    0035         MOVZWL  #FWA$S_NAMEBUF+-
                        0036                 FWA$S_TYPEBUF+FWA$S_VERBUF,-
0170 CA 012E 8F         0036                 FWA$Q_NAME(R10)                   ; setup result descriptor
             7E   7C    003C         CLRQ    -(SP)                             ; P5/P6 = 0
        0170 CA   9F    003E         PUSHAB  FWA$Q_NAME(R10)                   ; P4=addr. of rslt descriptor
          6C A9   9F    0042         PUSHAB  IFB$L_RNS_LEN(R9)                 ; P3=longword for length
        0188 CA   9F    0045         PUSHAB  FWA$Q_RNS(R10)                    ; P2=file name string descript.
       50    33   3C    0049         MOVZWL  #IO$_CREATE,R0                    ; ACP function code
```

```
              FFB1'  30  004C   168            BSBW    RM$FCPFNC                               ; call ACP and wait for reply
              0A 50  E8  004F   169            BLBS    R0,40$                                  ; branch if ok
                         0052   170            RMSERR  ENT,R1                                  ; set default error
              FFA6'  30  0057   171            BSBW    RM$MAPERR                               ; map STV status
                 16  11  005A   172            BRB     90$
                         005C   173
                         005C   174 ;
                         005C   175 ;          Return the RESULT NAME STRING to the user buffer
                         005C   176 ;
                         005C   177
              FFA1'  30  005C   178 40$:       BSBW    RM$COPY_RESULT                          ; copy RESULT NAME STRING
              10 50  E9  005F   179            BLBC    R0,90$                                  ; branch if error
                         0062   180
                         0062   181 ;
                         0062   182 ;          Return HIGHVER/LOWVER bits to caller in FNB
                         0062   183 ;
                         0062   184
                         0062   185            ASSUME  FIB$V_HIGHVER   EQ      FIB$V_LOWVER+1
                         0062   186
         02   0E     EF  0062   187            EXTZV   #FIB$V_LOWVER,#2,-                      ; extract the 2 bits
    50   0208 CA         0065   188            FWA$T_FIBBUF+FIB$W_NMCTL(R10),R0
                         0069   189
                         0069   190            ASSUME  NAM$V_HIGHVER   EQ      NAM$V_LOWVER+1
                         0069   191
    02   0E     50   F0  0069   192            INSV    R0,#NAM$V_LOWVER,#2,-                   ; store into FNB
              34 A7      006D   193            NAM$L_FNB(R7)
                         006F   194
           50   01   D0  006F   195            MOVL    #1,R0                                   ; success
              FF8B'  31  0072   196 90$:       BRW     RM$CLSCU                                ; evaporate IFAB and cleanup
                         0075   197
                         0075   198 ;
                         0075   199 ; Network specific $ENTER code
                         0075   200 ;
                         0075   201
                         0075   202 NTENTER:
              FF88'  30  0075   203            BSBW    NT$ENTER                                ; Do the $ENTER operation
              FF85'  31  0078   204            BRW     RM$CLSCU                                ; and clean up.
```

```
                     007B          .SBTTL  RM$RECOVER_FWA, Recover FWA from Expanded String
                     007B
                     007B    ;++
                     007B    ;
                     007B    ; RM$RECOVER_FWA - This routine uses the EXPANDED NAME STRING in the NAM
                     007B    ;                  block to restore the state of the FWA so that ACP
                     007B    ;                  operations can be performed using the FWA.
                     007B    ;
                     007B    ;                  Note that it is assumed that this is not a network
                     007B    ;                  operation, since the context is always kept in RMS
                     007B    ;                  space for search sequences.
                     007B    ;
                     007B    ; Inputs:
                     007B    ;
                     007B    ;       NAM$L_ESA        - Address of EXPANDED STRING BUFFER
                     007B    ;       NAM$B_ESL        - Length of EXPANDED STRING
                     007B    ;       NAM$W_FID/DID    - FID and DID of previous file
                     007B    ;       NAM$L_RSA        - Address of previous RESULT STRING
                     007B    ;       NAM$B_RSL        - Length of previous RESULT STRING
                     007B    ;       NAM$T_DVI        - Device name string
                     007B    ;
                     007B    ; Outputs:
                     007B    ;
                     007B    ;       R10 = FWA address
                     007B    ;
                     007B    ;       FWA$Q_NODE       - Is not altered.
                     007B    ;       FWA$Q_DEVICE     - Descriptor of device portion
                     007B    ;       FWA$Q_CDIR1      - Descriptor of entire root directory specification
                     007B    ;                            excluding the delimiters
                     007B    ;       FWA$B_ROOTERM    - Root directory terminator
                     007B    ;       FWA$Q_DIR1       - Descriptor of entire directory specification
                     007B    ;                            excluding the delimiters
                     007B    ;       FWA$B_DIRTERM    - Directory terminator
                     007B    ;       FWA$Q_RNS        - Descriptor of FILE NAME, TYPE and VERSION
                     007B    ;       FWA$B_DIRLEN     - Number of directory names present
                     007B    ;                          (needed for RM$COPY_RESULT)
                     007B    ;       FWA$Q_NAME+4     - Address of previous RESULT STRING
                     007B    ;       IFB$L_RNS_LEN    - Length of previous RESULT STRING
                     007B    ;
                     007B    ;       The FID and DID fields are copied into the FIB buffer
                     007B    ;       and a channel is assigned to the device.
                     007B    ;
                     007B    ;--
                     007B
                     007B    RM$RECOVER_FWA::
                     007B
                     007B    ;
                     007B    ;       Allocate FWA and temporary FSCB buffer
                     007B    ;
                     007B
       FF82'   30    007B           BSBW    RM$FWASET                       ; allocate FWA
       22 50   E9    007E           BLBC    R0,10$                          ; branch if error
52  0104 8F    3C    0081           MOVZWL  #FSCB$C_BLN,R2                  ; get size of FSCB
       FF77'   30    0086           BSBW    RM$GETSPC1                      ; allocate it
       17 50   E9    0089           BLBC    R0,10$                          ; branch if success
          5B   DD    008C           PUSHL   R11                             ; save impure area
          14   10    008E           BSBB    RECOVER_FWA                     ; do the work
       54     5B   D0  0090              MOVL    R11,R4                          ; get addr of buffer
              5B 8ED0  0093              POPL    R11                             ; restore impure area
              50   DD  0096              PUSHL   R0                              ; save status
  52    0104 8F    3C  0098              MOVZWL  #FSCB$C_BLN,R2                  ; get size
           FF60'   30  009D              BSBW    RM$RETSPC1                      ; return the space
              50 8ED0  00A0              POPL    R0                              ; restore status
                   05  00A3      10$:    RSB
                       00A4
                       00A4      ;
                       00A4      ;       Get descriptor of expanded name string from NAM block and probe it
                       00A4      ;
                       00A4
                       00A4      RECOVER_FWA:
       5B     51   D0  00A4              MOVL    R1,R11                          ; get address of FSCB
  56      0B A7    9A  00A7              MOVZBL  NAM$B_ESL(R7),R6                ; get length of expanded string
              64   13  00AB              BEQL    ERRESL                          ; error if none
  57      0C A7    D0  00AD              MOVL    NAM$L_ESA(R7),R7                ; get addr. of expanded string
                       00B1              IFNORD  R6,(R7),ERRESA                  ; error if cannot read buffer
   00000000'EF     16  00B7              JSB     RM$SCAN_STRING                  ; parse the string
  57      28 A8    D0  00BD              MOVL    FAB$L_NAM(R8),R7                ; restore name block
           FF3C'   30  00C1              BSBW    RM$CHKNAM                       ; still good?
           43 50   E9  00C4              BLBC    R0,25$                          ; nope
04 AB     0B A7    91  00C7              CMPB    NAM$B_ESL(R7),FSCB$Q_FILESPEC(R11); should be the same
              43   12  00CC              BNEQ    ERRESL
                       00CE
                       00CE      ;
                       00CE      ;       Copy device name to FWA.  If possible, get it from the expanded
                       00CE      ;       name string rather than the DVI to preserve the logical device name.
                       00CE      ;       Setup as concealed device however in order to use the DVI for the
                       00CE      ;       actual device assignment.
                       00CE      ;
                       00CE
  50    00E0 CA    7E  00CE              MOVAQ   FWA$Q_DEVICE(R10),R0            ; get addr to setup
    1D 6B     01   E1  00D3              BBC     #FSCB$V_DEVICE,(R11),10$        ; if not present, use DVI
60    14 AB   01   A3  00D7              SUBW3   #1,FSCB$Q_DEVICE(R11),(R0)      ; strip colon and stuff after it
        00E0 CA    B1  00DC              CMPW    FWA$Q_DEVICE(R10),-             ; name too long?
        00FF 8F        00E0                      #FWA$S_DEVICEBUF
              0F   1A  00E3              BGTRU   10$                             ; if too long, use DVI
              60   28  00E5              MOVC3   (R0),-                          ; copy to buffer
04 B0     18 BB        00E7                      @FSCB$Q_DEVICE+4(R11),@4(R0)
  50    00E8 CA    7E  00EB              MOVAQ   FWA$Q_CONCEAL_DEV(R10),R0       ; get addr to setup
                       00F0              SSB     #FWA$V_CONCEAL_DEV,(R10)        ; mark concealed dev present
                       00F4
    60    14 A7    9A  00F4      10$:    MOVZBL  NAM$T_DVI(R7),(R0)              ; get length of name
              11   13  00F8              BEQL    ERRDVI                          ; error if null
                       00FA
                       00FA              ASSUME  NAM$S_DVI  LE  FWA$S_DEVICEBUF
                       00FA
              60   28  00FA              MOVC3   (R0),-                          ; copy to buffer
04 B0     15 A7        00FC                      NAM$T_DVI+1(R7),@4(R0)
                       0100      20$:    SSB     #FWA$V_DEVICE,(R10)             ; mark device name present
           FEF9'   30  0104              BSBW    RM$ASSIGN                       ; assign channel to device
           13 50   E8  0107              BLBS    R0,STUFF_FID                    ; branch if ok
                   05  010A      25$:    RSB                                     ; exit with error
                       010B
                       010B      ERRDVI: RMSERR  DVI
                   05  0110              RSB
```

```
                              0111
                              0111  ERRESL: RMSERR  ESL
                         05   0116          RSB
                              0117
                              0117  ERRESA: RMSERR  ESA
                         05   011C          RSB
                              011D
                              011D  ;
                              011D  ;               Move FID, DID and WCC to FIB buffer
                              011D  ;
                              011D
                              011D                  ASSUME  FIB$W_DID          EQ      FIB$W_FID+6
                              011D                  ASSUME  FIB$L_WCC          EQ      FIB$W_DID+6
                              011D                  ASSUME  NAM$W_DID          EQ      NAM$W_FID+6
                              011D                  ASSUME  NAM$L_WCC          EQ      NAM$W_DID+6
                              011D
                              011D
                              011D  STUFF_FID:
     01F8 CA    24 A7    7D   011D                  MOVQ    NAM$W_FID(R7),FWA$T_FIBBUF+FIB$W_FID(R10)
     0200 CA    2C A7    7D   0123                  MOVQ    NAM$W_FID+8(R7),FWA$T_FIBBUF+FIB$W_FID+8(R10)
                              0129
                              0129  ;
                              0129  ; check for a rooted directory
                              0129  ;
                              0129
        5C 69    03      E1   0129                  BBC     #DEV$V_DIR,IFB$L_PRIM_DEV(R9),5$ ; if non-directory device
        2A 6B    02      E1   012D                  BBC     #FSCB$V_ROOT,(R11),1$            ; was there a root?
        50    1C AB      7D   0131                  MOVQ    FSCB$Q_ROOT(R11),R0              ; get descriptor
     0B AA    81 02      81   0135                  ADDB3   #2,(R1)+,FWA$B_ROOTERM(R10)      ; save root directory terminator
              50 03      C2   013A                  SUBL2   #3,R0                            ; remove directory terminators
        1A 69    1C      E1   013D                  BBC     #DEV$V_RND,IFB$L_PRIM_DEV(R9),1$; skip rooted directory if tape
      00FF 8F    50      B1   0141                  CMPW    R0,#FWA$C_MAXDIRLEN              ; root too big?
                 CF      1A   0146                  BGTRU   ERRESA                           ; error if too big
      00F0 CA    50      3C   0148                  MOVZWL  R0,FWA$Q_CDIR1(R10)              ; save length of directory
 00F4 DA    61   50      28   014D                  MOVC3   R0,(R1),@FWA$Q_CDIR1+4(R10)      ;  and save root dir. in FWA
                              0153                  SSB     #FWA$V_ROOT_DIR,(R10)            ; signal it as being there
                              0157                  SSB     #FWA$V_EXP_ROOT,(R10)            ;  and that it was explicit
                              015B
                              015B  ;
                              015B  ; Setup descriptor of the full directory spec
                              015B  ;
                              015B
        2A 6B    03      E1   015B  1$:             BBC     #FSCB$V_DIRECTORY,(R11),5$       ; branch if not found
        50    24 AB      7D   015F                  MOVQ    FSCB$Q_DIRECTORY(R11),R0         ; get descriptor
     0A AA    81 02      81   0163                  ADDB3   #2,(R1)+,FWA$B_DIRTERM(R10)      ; save directory terminator
              50 02      A2   0168                  SUBW2   #2,R0                            ; remove directory terminator
      00FF 8F    50      B1   016B                  CMPW    R0,#FWA$C_MAXDIRLEN              ; directory too big?
                 29      1A   0170                  BGTRU   9$                               ; error if too big
        13 69    1C      E1   0172                  BBC     #DEV$V_RND,IFB$L_PRIM_DEV(R9),5$; if tape, br to skip dir spec
      0130 CA    50      3C   0176                  MOVZWL  R0,FWA$Q_DIR1(R10)               ; save length of directory
 0134 DA    61   50      28   017B                  MOVC3   R0,(R1),@FWA$Q_DIR1+4(R10)       ; and save dir. in FWA
                              0181                  SSB     #FWA$V_DIR,(R10)                 ; mark directory present
        2E AA    01      90   0185                  MOVB    #1,FWA$B_DIRLEN(R10)             ; indicate 1 dir. name present
                              0189
        50    2C AB      7D   0189  5$:             MOVQ    FSCB$Q_NAME(R11),R0              ; set descriptor of file name
                 0F      12   018D                  BNEQ    10$
        50    3C AB      7D   018F                  MOVQ    FSCB$Q_VERSION(R11),R0           ; try file version
```

```
           0D   12 0193               BNEQ    20$
    50  34 AB   7D 0195               MOVQ    FSCB$Q_TYPE(R11),R0                 ; how about type?
           0B   12 0199               BNEQ    30$
         FF79   31 019B      9$:      BRW     ERRESA                              ; must be a name (or something)
    50  34 AB   A0 019E      10$:     ADDW2   FSCB$Q_TYPE(R11),R0                 ; add the type field
    50  3C AB   A0 01A2      20$:     ADDW2   FSCB$Q_VERSION(R11),R0              ; and the version
                   01A6
                   01A6      ;
                   01A6      ;        Use the EXPANDED NAME STRING as the pattern that the ACP should
                   01A6      ;        use for matching.  Note that this read-only buffer is actually
                   01A6      ;        in the user's (caller's) address space and thus cannot be
                   01A6      ;        written back by RMS or the ACP.
                   01A6      ;
                   01A6
0188 CA     50   B0 01A6      30$:     MOVW    R0,FWA$Q_RNS(R10)                   ; copy size (no flags)
018C CA     51   D0 01AB               MOVL    R1,FWA$Q_RNS+4(R10)                 ; copy address
                   01B0
                   01B0      ;
                   01B0      ;        Use the RESULT NAME STRING from the previous operation
                   01B0      ;        as input to the ACP so that it can recover its position
                   01B0      ;        within the directory.
                   01B0      ;
                   01B0
        6C A9   D4 01B0               CLRL    IFB$L_RNS_LEN(R9)                   ; preset length of prev. name
    56  03 A7   9A 01B3               MOVZBL  NAM$B_RSL(R7),R6                    ; get length of RESULT STRING
           45   13 01B7               BEQL    70$                                 ; branch if none
    57  04 A7   D0 01B9               MOVL    NAM$L_RSA(R7),R7                    ; get address of RESULT STRING
                   01BD               IFNORD  R6,(R7),ERRRSA                      ; error if cannot read buffer
  00000000'EF   16 01C3               JSB     RM$SCAN_STRING                      ; parse the string
    57  28 A8   D0 01C9               MOVL    FAB$L_NAM(R8),R7                    ; restore name block
        FE30'   30 01CD               BSBW    RM$CHKNAM                           ; still good?
        2E 50   E9 01D0               BLBC    R0,80$                              ; nope
 04 AB  03 A7   91 01D3               CMPB    NAM$B_RSL(R7),FSCB$Q_FILESPEC(R11); should be the same
           28   12 01D8               BNEQ    ERRRSA
    50  2C AB   7D 01DA               MOVQ    FSCB$Q_NAME(R11),R0                 ; get descriptor of file name
           0C   12 01DE               BNEQ    40$                                 ; add rest
    50  3C AB   7D 01E0               MOVQ    FSCB$Q_VERSION(R11),R0              ; get descriptor of file name
           0A   12 01E4               BNEQ    50$                                 ; add rest
    50  34 AB   7D 01E6               MOVQ    FSCB$Q_TYPE(R11),R0                 ; get descriptor of file name
           08   11 01EA               BRB     60$                                 ; copy name
                   01EC
    50  34 AB   A0 01EC      40$:     ADDW2   FSCB$Q_TYPE(R11),R0                 ; add the type field
    50  3C AB   A0 01F0      50$:     ADDW2   FSCB$Q_VERSION(R11),R0              ; and the version
    6C A9   50  3C 01F4      60$:     MOVZWL  R0,IFB$L_RNS_LEN(R9)                ; length of prev. RESULT NAME
0174 DA  61 50  28 01F8               MOVC3   R0,(R1),@FWA$Q_NAME+4(R10)          ; copy previous RESULT NAME
    50     01   D0 01FE      70$:     MOVL    #1,R0                               ; success
                05 0201      80$:     RSB
                   0202
                   0202      ERRRSA:  RMSERR  RST
                05 0207               RSB
                   0208
                   0208               .END
```

```
$$.PSECT_EP         = 00000000            NAM$L_ESA         = 0000000C
$$RMSTEST           = 0000001A            NAM$L_FNB         = 00000034
$$RMS_PBUGCHK       = 00000010            NAM$L_RSA         = 00000004
$$RMS_TBUGCHK       = 00000008            NAM$L_WCC         = 00000030
$$RMS_UMODE         = 00000004            NAM$S_DVI         = 00000010
DEV$V_DIR           = 00000003            NAM$T_DVI         = 00000014
DEV$V_RND           = 0000001C            NAM$V_HIGHVER     = 0000000F
ERRDVI                0000010B R      01  NAM$V_LOWVER      = 0000000E
ERRESA                00000117 R      01  NAM$V_NODE        = 00000011
ERRESL                00000111 R      01  NAM$W_DID         = 0000002A
ERRRSA                00000202 R      01  NAM$W_FID         = 00000024
FAB$L_NAM           = 00000028            NT$ENTER            *********  X    01
FIB$C_LENGTH        = 00000040            NTENTER             00000075 R      01
FIB$L_WCC           = 00000010            RECOVER_FWA         000000A4 R      01
FIB$V_HIGHVER       = 0000000F            RM$ASSIGN           *********  X    01
FIB$V_LOWVER        = 0000000E            RM$CHKNAM           *********  X    01
FIB$W_DID           = 0000000A            RM$CLSCU            *********  X    01
FIB$W_FID           = 00000004            RM$COPY_RESULT      *********  X    01
FIB$W_NMCTL         = 00000014            RM$ENTXAB_ARGS      00000000 RG     01
FIB$W_VERLIMIT      = 0000002C            RM$FCPFNC           *********  X    01
FSCB$C_BLN          = 00000104            RM$FSETI            *********  X    01
FSCB$Q_DEVICE       = 00000014            RM$FWASET           *********  X    01
FSCB$Q_DIRECTORY    = 00000024            RM$GETSPC1          *********  X    01
FSCB$Q_FILESPEC     = 00000004            RM$MAPERR           *********  X    01
FSCB$Q_NAME         = 0000002C            RM$RECOVER_FWA      0000007B RG     01
FSCB$Q_ROOT         = 0000001C            RM$RETSPC1          *********  X    01
FSCB$Q_TYPE         = 00000034            RM$SCAN_STRING      *********  X    01
FSCB$Q_VERSION      = 0000003C            RM$XAB_SCAN         *********  X    01
FSCB$V_DEVICE       = 00000001            RMS$ENTER         = 00000002 RG     01
FSCB$V_DIRECTORY    = 00000003            RMS$_DVI          = 000184F4
FSCB$V_ROOT         = 00000002            RMS$_ENT          = 0001C01A
FWA$B_DIRLEN        = 0000002E            RMS$_ESA          = 000184FC
FWA$B_DIRTERM       = 0000000A            RMS$_ESL          = 00018714
FWA$B_ROOTERM       = 0000000B            RMS$_RST          = 0001869C
FWA$C_MAXDIRLEN     = 000000FF            STUFF_FID           0000011D R      01
FWA$Q_CDIR1         = 000000F0            XAB$C_PRO         = 00000013
FWA$Q_CONCEAL_DEV   = 000000E8            XAB$C_PROLEN_V3   = 00000010
FWA$Q_DEVICE        = 000000E0            XBC$C_ENTPRO        *********  X    01
FWA$Q_DIR1          = 00000130
FWA$Q_FIB           = 00000010
FWA$Q_NAME          = 00000170
FWA$Q_RNS           = 00000188
FWA$S_DEVICEBUF     = 000000FF
FWA$S_NAMEBUF       = 00000100
FWA$S_TYPEBUF       = 00000028
FWA$S_VERBUF        = 00000006
FWA$T_FIBBUF        = 000001F4
FWA$V_CONCEAL_DEV   = 00000039
FWA$V_DEVICE        = 0000000F
FWA$V_DIR           = 0000000E
FWA$V_EXP_ROOT      = 0000003C
FWA$V_ROOT_DIR      = 0000003A
IFB$L_PRIM_DEV      = 00000000
IFB$L_RNS_LEN       = 0000006C
IO$_CREATE          = 00000033
NAM$B_ESL           = 0000000B
NAM$B_RSL           = 00000003
```

```
                                             +-----------------+
                                             ! Psect synopsis !
                                             +-----------------+

PSECT name                     Allocation           PSECT No.  Attributes
----------                     ----------           ---------  ----------
.  ABS  .                      00000000  (      0.)  00 (  0.)  NOPIC    USR    CON    ABS    LCL NOSHR NOEXE NORD   NOWRT NOVEC BYTE
RM$RMS                         00000208  (    520.)  01 (  1.)    PIC    USR    CON    REL    GBL NOSHR   EXE   RD   NOWRT NOVEC BYTE
$ABS$                          00000000  (      0.)  02 (  2.)  NOPIC    USR    CON    ABS    LCL NOSHR   EXE   RD     WRT NOVEC BYTE

                                          +------------------------+
                                          ! Performance indicators !
                                          +------------------------+

Phase                     Page faults    CPU Time       Elapsed Time
-----                     -----------    --------       ------------
Initialization                    36    00:00:00.10    00:00:00.55
Command processing               157    00:00:00.77    00:00:05.24
Pass 1                           430    00:00:16.53    00:00:39.52
Symbol table sort                  0    00:00:02.58    00:00:03.20
Pass 2                            86    00:00:02.88    00:00:05.91
Symbol table output               11    00:00:00.14    00:00:00.78
Psect synopsis output              2    00:00:00.03    00:00:00.05
Cross-reference output             0    00:00:00.00    00:00:00.00
Assembler run totals             724    00:00:23.03    00:00:55.25

The working set limit was 1650 pages.
92733 bytes (182 pages) of virtual memory were used to buffer the intermediate code.
There were 100 pages of symbol table space allocated to hold 1855 non-local and 22 local symbols.
428 source lines were read in Pass 1, producing 14 object records in Pass 2.
25 pages of virtual memory were used to define 24 macros.

                                         +--------------------------+
                                         ! Macro library statistics !
                                         +--------------------------+

Macro library name                       Macros defined
------------------                       --------------
_$255$DUA28:[RMS.OBJ]RMS.MLB;1                      12
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                       1
_$255$DUA28:[SYSLIB]STARLET.MLB;2                    7
TOTALS (all libraries)                              20

1977 GETS were required to define 20 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:RMS0ENTER/OBJ=OBJ$:RMS0ENTER MSRC$:RMS0ENTER/UPDATE=(ENH$:RMS0ENTER)+EXECML$/LIB+LIB$:RMS/LIB
```

0329 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY